International Journal of Information Technology, Modeling and Computing (IJITMC) Vol.9, No.1/2/3/4, November 2021

ON A PEDAGOGICAL ACTIVITIES SCHEDULING BASED ON THE JUGGLING THEORY

Telesphore Tiendrebeogo

Department of Mathematics and Computer Sciences, Nazi BONI University Bobo-Dioulasso, Burkina Faso

ABSTRACT

We are interested in a course scheduling strategy in academic backwardness context. Our aim is to find a course scheduling model that will solve the problem of delays in the context of insufficient infrastructure resources and teaching staff. We use juggling theory and multiprocessor scheduling to arrive at our approach. Thus, we have created a course scheduling algorithm that allows maximum use of available infrastructure and staff resources. Courses are scheduled in sequence by analogy to a juggling notation that describes the rhythm of throws and thus the objects trajectory in space in juggling (siteswap).

KEYWORDS

Real-time systems · Multiprocessor scheduling · juggling theory.

1. INTRODUCTION

The University is the pole par excellence of human capital development essential to the socio-economic development of any country. Most of the great personalities have assed through it and have developed and acquired their knowledge. Africa is not on the margins of this human development but faces innumerable obstacles. Thus, French-speaking African universities are the most affected and rank low in the world rankings of universities. There are several reasons for this:

– the study conditions are unfavorable;
– preventive measures are insufficient;
– the budgets allocated to universities are insufficient;
– the premature abandonment of students.

Most students prefer to compete in public and private sector competitions. Added to this are the insufficient number of teachers, the difficult research conditions and the low level of support for students and researchers. As far as infrastructures are concerned, they are insufficient, others are very old and the laboratories are almost non-existent or not equipped [2]. In the face of these obstacles, numerous meetings to discuss the issue of academic backwardness have not led to any concrete solution. Thus, in response to the call made by the Minister of Higher Education, Scientific Research and Innovation of Burkina Faso during the meeting of July 27 to 29, 2017 to find a solution to the academic backwardness, in this paper we propose a model for scheduling courses according to the infrastructure and teachers available like juggling.

2. MULTIPROCESSOR SCHEDULING AND JUGGLING THEORY

2.1. Principle

We are interested in scheduling policies capable of scheduling tasks on several processors on the one hand [3]-[9] and juggling theory on the other. Multi-processor scheduling is defined as a set of processors scheduling a given set of tasks. The study of this section will allow us to find a strategy that may relateto a set of teachers to run a given set of courses.

2.2. Multiprocessor Scheduling

The scheduling of a multiprocessor real-time system consists in defining a spatial and temporal allocation of work on several processors so that the time constraints are met. Indeed, single-processor scheduling aims to solve the problem of time allocation of the processor to tasks, while multi-processor scheduling adds a spatial allocation problem, i.e. which processor to use. Thus, the notion of migration, which consists of a switch from one processor to another executing the task, appears. There are three possible types of migration [3] :

- either no migration is possible (partitioned scheduling);
- either the migration is restricted to the work boundaries (or task migration);
- either migration is free: jobs can migrate during their execution (job migration).

2.3. Juggling Theory

1. Definition: Juggling, also known as juggling, is an exercise of skill consisting strictly of throwing, catching and throwing objects continuously into the air. It has been studied by great scientists. Among them, there is the American mathematician and engineer Claude SHANNON who is considered the father of information theory. He developed a theorem on juggling that bears his name.
2. SHANNON's Theorem: In the early 1980's, Claude SHANNON published the first formal mathematical juggling theorem giving the relationship between the amount of time juggling balls spend in the air and the time each ball spends in the juggler's hand. His theorem demonstrated the importance of the speed of the hands to succeed in juggling [10].

$$\mathbf{(f + d)H = (e + d)B}$$

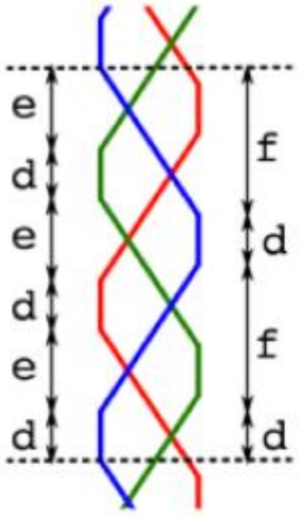


Figure 1. – Illustration du Théorème de SHANNON

The time of a period is counted in two ways: on the one hand for a ball (here the blue at right) and on the other hand for a hand (the left). We get : left period= 3(e + d) and right period= 2(f + d) and as the period is the same we have :(f + d)H = (e + d)B (1)

3. Our Approach

We propose a task scheduling model using some properties of the Juggling theory. We make a simple analogy between some elements of juggling with multiprocessor scheduling tools. More precisely we place ourselves in the context of multiprocessor scheduling using the global strategy. As a reminder, this strategy allows the migration of tasks from one processor to another.

3.1. Definitions and Terminology

1. Task: A task is a pedagogical activity to be carried out in order to achieve a well-defined result. We call a task system a finite set of tasks. τ = τ1, τ2 τ3 ... τn is a set of n tasks.
2. Work: A job is a part or subset of a task. In the multiprocessor context, the jobs of a task can be executed on different processors. In our case, a learning activity can be divided into sub-activities and performed by several people.
3. Processor: In the following we make an analogy between a person who has to perform a pedagogical task in an infrastructure and the processor.

3.2. Constraints

Let’s consider that at a given moment a processor can only execute one task, but that the tasks of a task can be done simultaneously on several processors (tutorials for several groups of students for example). As a reminder, our model wants to be as close as possible to a ”flexible real-time system”. This will only be possible if a processor resource (a teacher for example) is not available and a replacement is made as soon as possible in order to respect the deadline of the activity concerned.

4. MODELING

In this section, we propose a planning model. To begin, let us structure the fundamental entities of the study space. We consider the university environment as our study space. This space is made up of teachers without distinction, students, infrastructures, administrative and other personnel. We are interested in the first three and more particularly in the teachers and the infrastructures.

4.1. Teacher Modeling

They are the ones who carry out courses and other academic activities. By analogy to multiprocessor scheduling, we consider teachers as processor resources that have to perform tasks. From theSHANNON equation we have H which represents the number of hands. that juggle objects (balls for example). By analogy we consider a hand that juggles as a processor that performs a task and therefore the number of hands in the SHANNON equation substitutes forthe number of processors that perform the tasks. As each teacher has a very precise field of action i.e., his field of study and his discipline it is necessary to make a grouping of the teachers. Indeed, it would not be possible to pass the hand to any teacher. The latter will have to be authorized to carry out the requested task. Hence the importance of the grouping.

Let us define by E = E1, ..., En all teachers and Ef = Ef1 ., ..., Efn all teachers in a given subject area. We could thus have a view on the qualification of each one and if necessary make a judicious choice for the execution of a task. To be more rigorous we could go much deeper in the grouping in order to have groups of teachers by fields within the same stream. But we stop at the streams, given the insufficient number of teachers.

4.2. Infrastructure Modelling

Infrastructure is fundamental to the execution of activities. They are like memory spaces in which tasks are performed. That is why they must be organized in the same way as they are organizedfor teachers. We define I = I1 , ..., In the set of available infrastructures and If = If1 , ..., Ifn the set of infrastructures of a given sector. This does not mean that the infrastructures are partitioned for their sector only. We are in a context of insufficient resources so this optionis not possible. This rating just allows us to follow a logic to achieve our result.

5. ACTIVITY MODELING AND ALGORITHM

The educational activities are here the tasks intended to be executed by the processors (teachers). By analogy to juggling theory we consider a task to be performed as a juggling ball. It will be juggled until it reaches its time limit or deadline and will be taken out of the game. We limit ourselves to activities that involve the teaching couple infrastructure i.e., classes, tutorials and practical work that require the presence of a teacher and the use of a room or laboratory.

Let us first recall the SHANNON equation that we are going to use for the rest :

$$(f + d)H = (e + d)B \qquad (2)$$

With:

f = the time a ball spends in the air d = the time a ball passes in the hand H = the number of hands e = the time an empty hand spends B = the number of balls juggled.

Suppose that the time a ball stays in the air (f) corresponds to the time a task is being performed (because this time can be as long as possible, unlike d, which is the time a ball spends in a hand that can be neglected because, as soon as the ball returns to the hand, it is immediately thrown back). From the equation we can derive f -d(the time a ball spends in the air) $f = (e + d)B/H - d$ and we notice that this time is strongly dependent on the number of hands H and the number of balls B. The time f increases if the number of balls increases or if the number of hands decreases. On the contrary, if the number of handsincreases then the time f decreases which would lead to a faster executionof tasks in the multiprocessor context. We can neglect the time d that a ball passes through a hand which can be considered as the preemption time. Indeed, like a conservative scheduler, one can neglect the time that a hand remains empty as long as there are balls to juggle. As a reminder, a system is said to be conservative if it does not leave ready tasks waiting if a processor is available to run them, unlike an idle scheduler. Analyzing SHANNON's equation closely, we notice that all balls will always have the same flight time if their number and the number of hands remain constant. This would imply in our context to consider identical quantum of time. This would translate into siteswap notation by the cascade at f if f is odd or the fountain at f if f is even (f being the time the ball remains in the air), thus the time quantum.For example, if f = 3, then each class will run for 3 hours and pass the hand to the next and wait for its turn again.

Let's assume that as long as there are balls, the hands are just juggling. If a ball passes in one hand it is immediately thrown again, i.e., when a task is assigned to a processor it is automatically started, so the time that the ball passes in one hand is very small, for example d = 0.5. Similarly, the time that a hand remains empty will be very small (for example e = 0.5) since the objective is to perform a large number of tasks and since the balls come and go the hand cannot be empty. An empty hand would mean a processor with no tasks running.

With: $e = d = 0.5$ we have $f = (0.5 + 0.5)\ B/H - 0.5$ either $f = B/H - 0.5$. Knowing the time of the quantum f, we can calculate the time executed for a cycle, i.e., when all the balls pass for the first time. This time is $TempsExeCycle = B * f$. From this we can calculate the number of cycles it would take for a given time T;

$$NbCycle = T / TempsExeCycle \qquad (3)$$

Example: Let B = 10, H = 3; we have $10/3 - 0.5 = 2.83 \approx 3$. f = 3. Would mean that each ball must pass 3 units of time in the air for juggling to be possible. The time for a cycle is $TempsExeCycle = 10 * 3 = 30$. If the total duration of the task is T = 60 then NbCycle = 60/2 from where in two passes of the juggling the task will be finished.

The balls must be juggled by the hands, more concretely, in the academic context people (processors) must have the necessary skills to be able to pass the hand. For example, a history teacher will not be able to run an algebra class. In practice we can choose the value of f that suits us without going through the calculation since we are not going to juggle real balls. We can therefore do without this constraint. Siteswap rating is used at this level. Let us consider that siteswap is here a sequence of several quantum f of different tasks. For example, S = 2345 would mean that we have four (04) different courses, the first of which lasts 2 hours per week, the second 3 hours, the third 4 hours and the fourth 5 hours. Infrastructures are at the center of the modeling. There must be no discrepancies at their level, otherwise it will distort the whole program. So we need to give the number of infrastructures available for a given UFR or study path.

The length of the siteswap will then be checked to ensure that the length of the siteswap does not exceed the number of facilities available because each number in the siteswap will represent the time that a course will last in a room and assuming that all the courses represented by the siteswap are run at approximately the same time (morning or evening).

Let's consider C(D) = C1(D), C2(D), ..., Cn(D) the set of courses. Ci(D) represents the course i with its duration D. A sequence of courses to be performed will be for example S = C1(D)C2(D)C3(D).

Then for each class you have to list the different tasks to be performed that will correspond to the number of balls to be juggled. The next step is to calculate the duration of each task based on the number of staff members who are able to perform the tasks in the course and allocate each staff member one task. The last step will be to perform the tasks until the course is exhausted.

A course will be completed if all of its tasks are completed. The staff will be obliged to collaborate so that if necessary one in the group can take over the task (task migration). If a course ends it will be deleted from the C(D) course set, another in C(D) will replace it immediately. This is done until all the elements of the C(D) set, i.e. of the corresponding semester, have been used up. Note: Scheduled courses should be as independent as possible to avoid having staff (teachers) who may teach several scheduled courses at the same time. This will allow teachers to be less busy at the same time and therefore ready to take over a task. If a teacher is asked to teach at several levels when he can only teach in one place at a time, this will block the execution of the other classes andthus increase the delay. Each lesson is considered to have the following configuration:

Course =< (Entitled, TeachersNb, Duration, WeekDuration) > With:

- Title: represents the title of the course
- TeacherNb: represents the number of teachers in the course
- Duration: represents the course duration (hourly volume)
- WeekDuration: represents the weekly duration of the course.

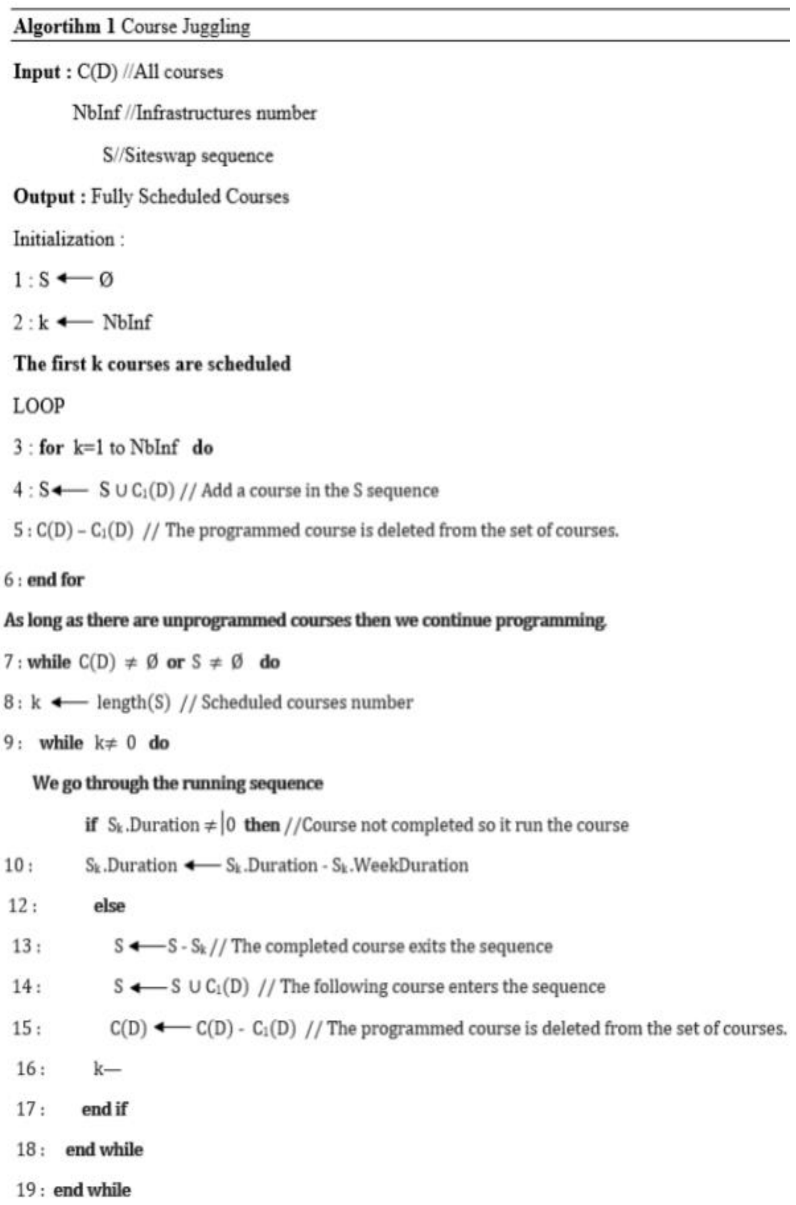
Based on all this information, we propose a lesson scheduling algorithm that relies heavily on juggling. Note that this algorithm will only work under well-defined conditions.

6. RESULT

In the Input part we have the input parameters C(D), NbInf and S. C(D) which represents the set of courses to be scheduled NbInf which represents the number of available infrastructures S which represents the sequence of courses in execution. In the first loop while the first sequence of lessons is programmed, the programmed lessons are deleted from C(D).

At the second while loop, as long as there are courses in C(D) or in Running and programming of the courses continues until both sets are empty. Knowing that the initial length of S is the number infrastructures, then there is no risk of overflowing into the loop since this length can never exceed the number of infrastructures. The execution here consists in decrementing (subtracting) from the remaining duration of each programmed course its weekly duration.
In reality this will correspond to the number of days the course is running.

It is also assumed that during the day there are two (02) sequences (one for the morning and one for the evening). The choice of weekly schedules must be such that a room cannot be free during a sequence of classes. For example, you should avoid programs such as 08h-10h which will leave 10h-12h free while there are pending classes. To remedy this, the weekly schedules should be as large as possible or make sure that there is one class that replaces another that finishes early.



Algortihm 1 Course Juggling

```
Input : C(D) //All courses
        NbInf //Infrastructures number
          S//Siteswap sequence
Output : Fully Scheduled Courses
Initialization :
1 : S ⟵ Ø
2 : k ⟵ NbInf
The first k courses are scheduled
LOOP
3 : for k=1 to NbInf do
4 : S⟵ S ∪ $C_1(D)$ // Add a course in the S sequence
5 : C(D) – $C_1(D)$ // The programmed course is deleted from the set of courses.
6 : end for
As long as there are unprogrammed courses then we continue programming
7 : while C(D) ≠ Ø or S ≠ Ø do
8 : k ⟵ length(S) // Scheduled courses number
9 :   while k≠ 0 do
    We go through the running sequence
          if $S_k$.Duration ≠ |0 then //Course not completed so it run the course
10 :      $S_k$.Duration ⟵ $S_k$.Duration - $S_k$.WeekDuration
12 :       else
13 :          S ⟵S - $S_k$ // The completed course exits the sequence
14 :          S ⟵ S ∪ $C_1(D)$ // The following course enters the sequence
15 :          C(D) ⟵ C(D) - $C_1(D)$ // The programmed course is deleted from the set of courses.
16 :       k—
17 :     end if
18 :   end while
19 : end while
```

7. Conclusion

The model we have proposed is intended to be a collaborative model for achieving objectives. Moreover, multiprocessor scheduling, which implies the collaboration of several processors to schedule several tasks, is in the same direction. Our course scheduling model is based on these two broad areas. The limitation of our model is that for the moment it does not take into account the latest end date of a course, so we do not go into the details of the course schedule, i.e., who runs which course and in which room. We have limited ourselves in this document to a global model for the scheduling of courses in a given course, course of study or training and research unit (TRU).A significant amount of work remains to be done. First define another objective function thatwill minimize off-peak hours in the schedule. Then the extension of your work with more data andconstraints.

REFERENCES

[1] L. Dit Menza, " La théorie mathématique de la jonglerie. la quadrature de la balle", inLaboratoire de Mathématiques, URCA, 2015.
[2] Direction des ´etudes et de la Planification, " Tableau de bord de l'enseignement superieur,annee scolaire 2011/2012", in Tableau de bord de l'enseignement supérieur, annéescolaire 2011/2012, 2013.
[3] J. Carpenter and S. Funk, " A categorization of real-time multiprocessor schedulingproblems and algorithms," in Handbook on Scheduling Algorithms, Methods, and Models,2004, p. 30.
[4] P. Regnier, G. Lima, and E. Massa, " Run: Optimalmultiprocessor real- time scheduling via reduction to uniprocessor," in Proceedings of the 32nd IEEE Real-Time SystemsSymposium (RTSS), 2011, p. 104–115.
[5] E. Massa and G. Lima, " Optimal and adaptivemultiprocessor real-time scheduling:The quasi-partitioning approach," in Proceedings of the 26th Euromicro Conference onReal-Time Systems (ECRTS), 2014.
[6] J. M. Calandrino and J. H. Anderson, " A hybrid realtime scheduling approach for large-scale multicore platforms," in Proceedings of the 19th Euromicro Conference on Real-TimeSystems (ECRTS), 2007, p. 247–258.
[7] J. Goossens and S. Funk, " Edf scheduling on multiprocessor platforms : some (perhaps)counterintuitive observations," in Proceedings of the International Conference on Real-Time Computing Systems and Applications (RTCSA), 2002.
[8] J. Goossens,S. Funk and S. Baruah " Priority-driven scheduling of periodic task systemson multipro-cessors," in Real-Time Systems, 2003, p. 187–205.
[9] S. Baruah, " Optimal utilization bounds for the fixed-priority scheduling of periodic tasksystems on identicalmultiprocessors," in IEEE Trans- actions on Computers, 2004, p. 781– 784.
[10] R. Graham and J. Buhler, " L'art de jongler," in La recherche N135 Juillet/Aoˆut, 1982.

AUTHORS

Telesphore Tiendrebeogo PhD and overlay network and assistant professor at Nazi Boni University. I have a master's degree in multimedia and real time system. My current research is on big data and image watermarking.